UNDERSTANDING KNOWLEDGE ACQUISITION AND RELEASE IN LANGUAGE MODELS VIA CIRCUITS

Kiran Raja[1], Arav Maheria[2], Andrew Bae[3], Alan Sun[4]
[1]Alcuin School, [2]Brown University, [3]Columbia University, [4]Carnegie Mellon University
{kiran.rajatx1, andrew.bae08}@gmail.com, arav_maheria@brown.edu, awsun@cmu.edu

ABSTRACT

General agents must acquire new capabilities while preserving existing ones. Two phenomena make this balance difficult: grokking, where memorization abruptly ends during training; and forgetting, where previously learned skills rapidly degrade under sequential learning. Although both are typically studied in isolation, we argue that they admit a unified mechanistic explanation. For a fixed task, we hypothesize that grokking and forgetting occur precisely when the stability of a model's circuits increases and decreases across subtasks, respectively. Through a case study of `Llama-3.2-1B` across tasks such as factual retrieval, logical and commonsense reasoning, as well as bias evaluation, we find evidence supporting this hypothesis. To our knowledge, this is the first architecture- and task-agnostic measure for grokking and forgetting. Our results suggest that by leveraging mechanistic insights, generalization phase transitions can be measured directly on the training set.[1]

1 INTRODUCTION

As language-model agents become prevalent, they must be able to continuously learn and adapt. At the same time, we expect their core linguistic and reasoning capabilities to stay consistent. Balancing this plasticity-stability tradeoff is one challenge to building general agents. In this paper, we study two seemingly disparate phenomena that disrupt this balance: **grokking** and **forgetting**.

A model **groks** when it initially memorizes the training set, then undergoes a phase transition where generalization error decreases sharply (Power et al., 2022; Nanda et al., 2023; Miller et al., 2024, *inter alia*). The time between the memorization and generalization phase is unpredictable and difficult to measure. For practitioners finding this sweet spot is crucial: stopping too early may lead to poor generalization, while stopping too late wastes compute. However, developing a precise, task- and model-independent statistic for grokking has remained elusive. On the other hand, when a model learns a sequence of tasks it can undergo **forgetting**: performance on early tasks degrades as subsequent tasks are learned. When this degradation occurs rapidly, we term this *catastrophic* (McCloskey and Cohen, 1989; Kirkpatrick et al., 2017; Delange et al., 2021, *inter alia*). Mitigating forgetting remains a central challenge to continual learning. To this end, these phenomena represent extreme regimes where the plasticity-stability relation breaks down. Understanding when and how they occur give insight into necessary conditions for general, adaptable agents.[2]

Grokking and forgetting are typically considered distinct, independent phenomena and studied as such. Inspired by recent progress in mechanistic interpretability, we propose a unified perspective by studying a model's **circuit** and its evolution during training. For a fixed task, a model's circuit is a (minimal) subset of the model's components that drive its functional behavior (Elhage et al., 2021; Wang et al., 2022). In contrast to many existing mechanistic approaches, we treat circuits as first-class objects and entirely divorce them from their interpretations (Tigges et al., 2024; Jiang et al., 2025, *inter alia*). In particular, we adopt an approach from Sun (2025) that formalizes and measures **circuit stability**: the extent to which a model consistently applies same circuit across subsets of inputs. *We hypothesize that grokking and (catastrophic) forgetting occur precisely when circuit*

[1]Our codebase is available at https://anonymous.4open.science/r/forgot-200F/.
[2]We defer a detailed discussion of the related literature to Appendix A.

stability rapidly increases and decreases, respectively. Through a case study of `Llama-3.2-1B` on a diverse set of tasks—including knowledge retrieval, logical and common-sense reasoning, as well as bias evaluation—we provide evidence for our circuit stability hypothesis.

2 BACKGROUND

Herein, we briefly introduce circuit stability and refer readers to Sun (2025) for a detailed treatment. Let $\mathcal{X}, \mathcal{Y}$ be input and output spaces, respectively. We view a neural network $f : \mathcal{X} \to \mathcal{Y}$ through its computation graph $\mathcal{G}_f = (\mathcal{V}_f, \mathcal{E}_f)$ where $\mathcal{V}_f, \mathcal{E}_f$ are vertices representing attention heads in f and edges denoting computational dependencies between vertices. Let $\mathcal{D}_{\mathcal{X}\times\mathcal{Y}}$ be a data distribution and consider a partition of $\mathcal{X} \times \mathcal{Y}$, $\mathcal{S}$. Also, let $L : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}^{\geq 0}$ be some loss function.

Definition 2.1. Let $S \subset \mathcal{X} \times \mathcal{Y}$ and let $\mathcal{D}_S$ denote the data distribution restricted to S. The **circuit** of f on S is the map $c : \mathcal{E}_f \to \mathbb{R}$. For each $e \in \mathcal{E}_f$,

$$c(e) \triangleq \mathbb{E}_{(x,y),(x',\cdot)\sim\mathcal{D}_S}\left[L\big(f_{e(x)\leftarrow e(x')}(x), y\big) - L(f(x), y)\right].$$

Here $e(x)$ denotes the value of component e when f is evaluated on input x, $f_{e(x)\leftarrow e(x')}(x)$ is $f(x)$ under the intervention that replaces $e(x)$ with $e(x')$ obtained from input x'.

We view S as a **subtask** of the task $\mathcal{D}_{\mathcal{X}\times\mathcal{Y}}$. In this way, a circuit may be thought of as an embedding of subtasks. In practice, for the sake of efficiency, we compute Definition 2.1 with a first-order Taylor approximation (Hanna et al., 2024; Kramár et al., 2024):

$$c_S(e) \approx \mathbb{E}_{(x,y),(x',\cdot)\sim\mathcal{D}_S}[(e(x') - e(x))^\top \nabla_{e(x)} L(f(x), y)].$$

We now define circuit stability as the smoothness of this induced subtask embedding space.

Definition 2.2. Let $t : \mathcal{X} \times \mathcal{Y} \to \mathcal{S}$ where $t(x, y) \mapsto S$ if and only if $(x, y) \in S$. Denote $t_\star \mathcal{D}_{\mathcal{X}\times\mathcal{Y}}$ the pushforward distribution under t. f's **circuit stability** on $\mathcal{S}$ is

$$\text{stable}(f, \mathcal{S}) \triangleq \mathbb{E}_{S,S'\sim t_\star\mathcal{D}_{\mathcal{X}\times\mathcal{Y}}} K(c_S, c_{S'}),$$

where K is an RKHS kernel. We consistently take K to be the rank correlation between $c_S, c_{S'}$.

3 GROKKING AS CIRCUIT STABILIZATION

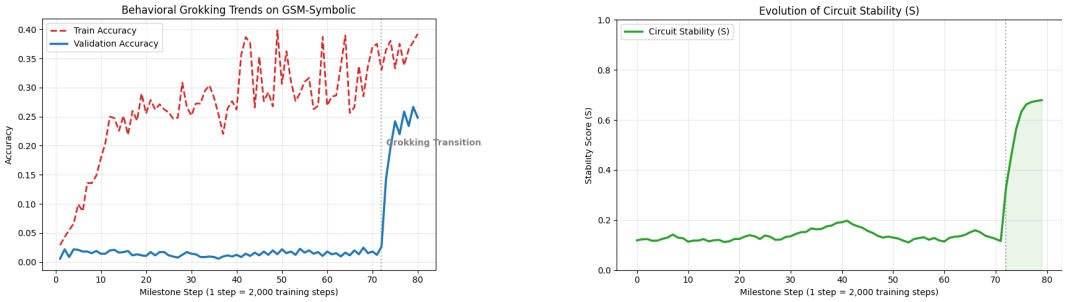
Nanda et al. (2023) show for small Transformers trained on modular arithmetic, grokking occurs from "algorithmic stabilization:" when model's learned algorithm stabilizes around a generalizing one. In similar vein, Tigges et al. (2024) show across model scales on toy tasks, a model's circuits evolve over the training horizon. Building on these observations, we measure circuit stability during instruction-tuning. Specifically, we hypothesize that the model transitions between the memorization (high training accuracy and generalization error) and generalization (high training accuracy and low generalization error) regime when circuit stability undergoes a sharp phase transition from low to high. Since our approach does not require understanding the algorithm behavior of the extracted circuits[3] we can in principle measure this for arbitrarily large models and complex tasks.

In this case, we choose `GSM-Symbolic` as it combines symbolic reasoning with mathematical and logical reasoning (Mirzadeh et al., 2025). We instruction-tuned `Llama-3.2-1B` and evaluate both performance (measured in accuracy) as well as circuit stability every 2,000 steps.[4] Our results are shown in Figure 1.

Memorization Phase. During the initial 70 milestones (1.4M steps), the model's generalization gap widens. While training accuracy increases gradually to 0.35, validation accuracy remains near zero. In this phase, Circuit stability is low and inconsistent, fluctuating between 0 and 0.1 (see Figure 1a). To this end, we expect the model to use input-specific heuristics or "noisy" circuits on semantically similar subtasks that potentially require the same reasoning process.

[3] This being the primary objective of mechanistic interpretability.

[4] Because of memory constraints, we use a batch size of 2 with no gradient accumulation (Marek et al., 2025). For brevity, we refer to each one of these measurements as **milestones**. We choose subtasks by partitioning the input space by the first letter in each prompt. This induces a true partition while also keeping cells in the partition semantically similar.



(a) (b)

Figure 1: (a) shows grokking. After training accuracy saturates at around milestone 40, validation continues to plateau at 0. The phase transition from memorization to generalization occurs at milestone 72. (b) shows circuit stability during training. Stability increases sharply at the same step. Note that since circuit stability is computed as an expected rank correlation, it is bounded in $[-1, 1]$.

Stability-Accuracy Alignment. At milestone 72 (step 1.42M), we observe a simultaneous surge in validation accuracy and Circuit Stability. Validation accuracy rises abruptly to 0.38 from 0.05 (see gray dotted line in Figure 1b). Synchronized with this behavioral shift, circuit stability increases from a mean of 0.15 to a peak of 0.70. To further qualify the relationship between circuit stability and generalization error, we compute the correlation between stability and validation accuracy throughout the training sequence. We find a strong positive correlation ($r = 0.87$, $p < 0.001$), supporting the hypothesis that circuit stabilization is the underlying mechanistic driver of grokking.

4 Forgetting as Circuit Decay

We now turn our attention to forgetting. Forgetting has been extensive studied in weight and representation spaces, and most recently also mechanistically (Kirkpatrick et al., 2017; Rolnick et al., 2019; Jiang et al., 2025, *inter alia*). In one line of work, Todd et al. (2024) show the existence of **function vectors** (FVs): task-specific directions in a model's activation space, when injected at multiple intermediate layers induce the model to perform the corresponding function without additional gradient updates. Building on this idea, Jiang et al. (2025) remarkably demonstrate that forgetting occurs when a model's FV difts. Herein, we extend this mechanistic framework and demonstrate that circuit stability is a better predictor of forgetting in three ways:

1. Circuit stability is a strictly stronger criterion than FV stability. That is, circuit stability implies FV stability, but the converse does not hold (see Proposition 4.1).
2. Circuit stability is correlated with forgetting in cases where as a model's FV may stay in tact.
3. Based on these insights, we introduce stability-guided regularization and replay to mitigate forgetting.

We first show that circuit stability implies FV stability and that the converse does not hold. Our result applies to single-layer models and assumes that all edges of the computation graph are a part of its circuit (Section 2). This captures the core intuition and by inducting over the number of layers with appropriate masking, the result can be easily extended to multi-layer models with sparsely distributed circuits.

Proposition 4.1. *Consider a time-indexed (over the training horizon) model $f_t : \mathcal{X} \to \mathbb{R}^d$ with computation graph $(\mathcal{V}_{f_t}, \mathcal{E}_{f_t})$. Assume that f_t admits an **additive residual decomposition**:*

$$f_t : x \mapsto \sum_{v \in \mathcal{V}_{f_t}} r_v^{(t)}(x). \tag{1}$$

Let $\mathcal{D}_0, \mathcal{D}_1$ be baseline and tasks distributions (respectively) over $\mathcal{X} \times \mathbb{R}^d$ (Todd et al., 2024). Let $v^{(t)}$ be the FV at t such that

$$v^{(t)} \triangleq \mathbb{E}_{x \sim \mathcal{D}_1, x' \sim \mathcal{D}_0}[\, f_t(x) - f_t(x')]. \tag{2}$$

*Also, define $(\varepsilon, \mathcal{D}_0, \mathcal{D}_1)$-**circuit drift** between times t and t' by*

$$\Delta(t, t')^2 \triangleq \sum_{v \in \mathcal{V}_{f_t}} \left(\mathbb{E}_{x \sim \mathcal{D}_1, x' \sim \mathcal{D}_0} \left[\|f_t(x) - f_{t'}(x)\|_2^2 \| + \|f_t(x') - f_{t'}(x')\|_2^2 \right] \right). \tag{3}$$

		Sequence 1		
Task	t_1	t_2	t_3	t_4
BoolQ	0.13 / 0.62	0.09 / 0.00	0.05 / 0.00	0.11 / 0.00
MMLU-CS	0.15 / ✗	0.34 / 0.56	0.21 / 0.11	0.13 / 0.00
Winogrande	0.16 / ✗	0.16 / ✗	0.18 / 0.5	0.15 / 0.05
StereoSet	0.18 / ✗	0.17 / ✗	0.18 / ✗	0.22 / 0.33
		Sequence 2		
Task	t_1	t_2	t_3	t_4
StereoSet	0.21 / 0.33	0.11 / 0.05	0.12 / 0.00	0.08 / 0.00
Winogrande	0.15 / ✗	0.20 / 0.51	0.16 / 0.45	0.13 / 0.04
MMLU-CS	0.18 / ✗	0.22 / ✗	0.37 / 0.56	0.30 / 0.11
BoolQ	0.08 / ✗	0.09 / ✗	0.11 / ✗	0.13 / 0.61

Table 1: For each sequence of learned tasks, at a given time t_i we measure the model's accuracy (a) and it's circuit stability (Δ). These are organized as (Δ / a) in each cell. We do not evaluate the accuracy of a model before it is trained. This is denoted by ✗.

1. If $\Delta(t, t') \leq \varepsilon$, then

$$\|v^{(t)} - v^{(t')}\|_2 \lesssim \epsilon\sqrt{|\mathcal{V}_{f_t}|}. \tag{4}$$

2. There exist $(f_t, f_{t'})$ such that $v^{(t)} = v^{(t')}$ but $\Delta(t, t')$ can be made arbitrarily large.

The proof is presented in Appendix B. Here, circuit drift is a dissimilarity counterpart to circuit stability. Driven the insights from Proposition 4.1 and evidence that circuit stability is tied to grokking, we empirically show that circuit stability is also correlated with forgetting during sequential learning. We design a curriculum below and measure accuracy, circuit stability, and FV drift as a function of time.

Curriculum. We take $t \in [0, 4]$ and finetune `Llama-3.2-1B` on four tasks (with $t = 0$ corresponding to the pre-trained model): BoolQ, MMLU, Winogrande, and StereoSet (Clark et al., 2019; Hendrycks et al., 2021; Sakaguchi et al., 2021; Nadeem et al., 2021). We consider two sequences of tasks, sequence 1 follows the original ordering above and sequence 2 is the reverse. We train one epoch per task.[5] For both sequences, we use the same optimizer hyperparameters. Checkpoints and measurements are taken at the end of each epoch. Further experimental details are provided in Appendix C.

Immediate vs. Protracted Decay. Table 1 demonstrates that catastrophic forgetting is characterized by a simultaneous collapse in task accuracy and circuit stability. Firstly, across all tasks in both sequences we observe a positive correlation between accuracy and circuit stability. As forgetting occurs and accuracy decreases monotonically with time, we also see that circuit stability is decreasing. Interestingly, for each task we observe **distinct circuit stability decay rates**. For example, logical reasoning (BoolQ) exhibits immediate an accuracy drop $0.62 \to 0.00$ when transitioning to MMLU. This is accompanied by a drop in stability $0.13 \to 0.09$ (-30%). Conversely, knowledge-retrieval tasks like MMLU show a gradual decline in accuracy and circuit stability. Since all tasks we benchmark require some learned knowledge, we suspect that the associative pathways (i.e. key-value retrieval mechanisms) are preserved even as task-specified memories become distorted (Geva et al., 2021; Meng et al., 2022). In this way, circuit stability provides a task-agnostic early warning sign for forgetting. Inspired by this, we show next that circuit stability can be used to construct an effective replay buffer of examples.

5 Disentangling Latent Triggers and Circuit Stability

We now seek to demonstrate that a loss in circuit stability is sufficient for forgetting. To do this, we need to decouple representation and circuit drift. During standard fine-tuning representations and circuits are entangled: as the model learns it adapts its previously learned representations while simultaneously rewiring its causal pathways (Geiger et al., 2024; Wadhwa et al., 2025). In this case, a drop accuracy is ambiguous it would reflect a structural fragmentation of the circuit or a drift in

[5]Throughout, we use batch size 1. Marek et al. (2025) demonstrate that this is principled. To make finetuning on MMLU tractable, we only consider the computer science split (CS).

Table 2: **FV–circuit ablation: accuracy, FV drift, and circuit stability under two intervention settings.** Preserving the FV while allowing circuit fragmentation yields collapse despite low drift; enforcing stability preserves function despite comparable drift.

Setting	Accuracy				Stability (S)		
	t_1	t_2	t_3	t_4	Initial	Final	Change
Drifted Circuit							
BoolQ	0.62	0.00	0.00	0.00	0.53	0.22	-59%
MMLU	✗	0.56	0.11	0.11	0.26	0.081	-69%
Winogrande	✗	✗	0.50	0.50	0.93	0.21	-77%
Stereoset	✗	✗	✗	0.33	-0.018	-0.085	-470%
Fixed Circuit							
BoolQ	0.62	0.24	0.03	0.00	0.24	0.21	-14%
MMLU	✗	0.56	0.17	0.11	0.22	0.20	-8%
Winogrande	✗	✗	0.5043	0.50	0.88	0.88	$+0\%$
Stereoset	✗	✗	✗	0.34	0.24	0.21	-14%

Table 3: *(Top)*Accuracy and circuit stability when circuit is allowed to drift but function vector is held still. *(Bottom)*Same measurements except function vector is allowed to drift while circuit is structurally fixed.

the representations. Herein, we pose a sharper question: *what if we preserve the representation and allow the circuit to drift and vice versa? Does performance persist?*

To show that a loss of circuit stability is sufficient for forgetting, we decouple *representation drift* from *circuit drift*. This separation is necessary because standard fine-tuning entangles the two: as the model trains on a new task, task-specific activations (the *latent trigger*) can move in representation space while the underlying computation graph simultaneously rewires. Under this entanglement, a drop in accuracy is ambiguous—it could reflect loss of the trigger, fragmentation of the circuit that implements the procedure, or both. We therefore ask a sharper causal question: if we *preserve* the trigger but allow the circuit to change, does the model still forget? And conversely, if we allow the trigger to drift but *preserve* the circuit structure, does performance persist?

We keep the same experimental setup as before. We add to the fine-tuning objective two different forms of regularization which we describe briefly here and detail in Appendix H:

1. **Fixed FV, Drifted Circuit.** We use forward-pass hooks continuously patch in the cached FV from their initial state. This prevents the FV from drifting while the circuit is allowed to vary.
2. **Drifted FV, Fixed Circuit.** We enforce structural circuit stability using a structure-constrained similar to Gupta et al. (2024) objective.

Impact of Circuit Drift. Our results in Table 3 show that preserving the functional representation is insufficient for retaining task performance. In the drifted circuit setting, BoolQ accuracy collapses $0.62 \rightarrow 0.0$ when training on the next dataset. This mirrors our observations above, where a loss in accuracy is coupled with a loss in circuit stability. Conversely, the model maintains its performance when circuit stability is enforced, even under significant representation drift. In this setting, we see that performance decay is more gradual and in some cases (Winogrande) non-existent.

In Appendix I, we present a simple procedure based on circuit stability that caches examples which exhibit high stability during initial training and replays them during subsequent learning. Like our fixed circuit regulation above, we find that this mitigates forgetting.

6 Conclusion

In this paper, we propose circuit stability as a structural metric linking knowledge acquisition (grokking) and release (forgetting). Our case study on `Llama-3.2-1B` shows that circuit stability tracks generalization error across diverse tasks, rising during grokking and declining before performance collapse. Based on these insights, we develop stability-guided regularization and replay strategies that preserve task-relevant reasoning circuits. Future work will extend these mechanistic analyses to larger models and longer task sequences.

ACKNOWLEDGMENTS

We would like to thank Lambda for their support in providing credits for access to GPU instances. Alan is supported by the National Science Foundation Graduate Research Fellowship Program under Grant No. DGE2140739.

REFERENCES

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2924–2936, Minneapolis, Minnesota. Association for Computational Linguistics.

Matthias Delange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Ales Leonardis, Greg Slabaugh, and Tinne Tuytelaars. 2021. A continual learning survey: Defying forgetting in classification tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, page 1–1.

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Nova DasSarma, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. 2021. A mathematical framework for transformer circuits. *Transformer Circuits Thread*.

Atticus Geiger, Zhengxuan Wu, Christopher Potts, Thomas Icard, and Noah Goodman. 2024. Finding alignments between interpretable causal variables and distributed neural representations. In *Proceedings of the Third Conference on Causal Learning and Reasoning*, volume 236 of *Proceedings of Machine Learning Research*, pages 160–187. PMLR.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. 2021. Transformer feed-forward layers are key-value memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5484–5495, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Rohan Gupta, Iván Arcuschin, Thomas Kwa, and Adrià Garriga-Alonso. 2024. Interpbench: Semi-synthetic transformers for evaluating mechanistic interpretability techniques. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Michael Hanna, Sandro Pezzelle, and Yonatan Belinkov. 2024. Have faith in faithfulness: Going beyond circuit overlap when finding model mechanisms. In *First Conference on Language Modeling*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. In *International Conference on Learning Representations*.

Gangwei Jiang, Caigao JIANG, Zhaoyi Li, Siqiao Xue, JUN ZHOU, Linqi Song, Defu Lian, and Ying Wei. 2025. Unlocking the power of function vectors for characterizing and mitigating catastrophic forgetting in continual instruction tuning. In *The Thirteenth International Conference on Learning Representations*.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526.

János Kramár, Tom Lieberum, Rohin Shah, and Neel Nanda. 2024. Atp*: An efficient and scalable method for localizing llm behaviour to components. *Preprint*, arXiv:2403.00745.

Ziming Liu, Eric J Michaud, and Max Tegmark. 2023. Omnigrok: Grokking beyond algorithmic data. In *The Eleventh International Conference on Learning Representations*.

Martin Marek, Sanae Lotfi, Aditya Somasundaram, Andrew Gordon Wilson, and Micah Goldblum. 2025. Small batch size training for language models: When vanilla SGD works, and why gradient accumulation is wasteful. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Michael McCloskey and Neal J. Cohen. 1989. Catastrophic interference in connectionist networks: The sequential learning problem. In Gordon H. Bower, editor, *Psychology of Learning and Motivation*, volume 24, pages 109–165. Academic Press.

Kevin Meng, David Bau, Alex J Andonian, and Yonatan Belinkov. 2022. Locating and editing factual associations in GPT. In *Advances in Neural Information Processing Systems*.

Jack William Miller, Charles O'Neill, and Thang D Bui. 2024. Grokking beyond neural networks: An empirical exploration with model complexity. *Transactions on Machine Learning Research*.

Seyed Iman Mirzadeh, Keivan Alizadeh, Hooman Shahrokhi, Oncel Tuzel, Samy Bengio, and Mehrdad Farajtabar. 2025. GSM-symbolic: Understanding the limitations of mathematical reasoning in large language models. In *The Thirteenth International Conference on Learning Representations*.

Moin Nadeem, Anna Bethke, and Siva Reddy. 2021. StereoSet: Measuring stereotypical bias in pretrained language models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5356–5371, Online. Association for Computational Linguistics.

Neel Nanda, Lawrence Chan, Tom Lieberum, Jess Smith, and Jacob Steinhardt. 2023. Progress measures for grokking via mechanistic interpretability. *Preprint*, arXiv:2301.05217.

Alethea Power, Yuri Burda, Harri Edwards, Igor Babuschkin, and Vedant Misra. 2022. Grokking: Generalization beyond overfitting on small algorithmic datasets. *Preprint*, arXiv:2201.02177.

David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy P. Lillicrap, and Greg Wayne. 2019. Experience replay for continual learning. *Preprint*, arXiv:1811.11682.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. Winogrande: an adversarial winograd schema challenge at scale. *Commun. ACM*, 64(9):99–106.

Alan Sun. 2025. Circuit stability characterizes language model generalization. In *The 63rd Annual Meeting of the Association for Computational Linguistics*.

Curt Tigges, Michael Hanna, Qinan Yu, and Stella Biderman. 2024. LLM circuit analyses are consistent across training and scale. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Eric Todd, Millicent Li, Arnab Sen Sharma, Aaron Mueller, Byron C Wallace, and David Bau. 2024. Function vectors in large language models. In *The Twelfth International Conference on Learning Representations*.

Somin Wadhwa, Silvio Amir, and Byron C. Wallace. 2025. Circuit distillation. *Preprint*, arXiv:2509.25002.

Kevin Wang, Alexandre Variengien, Arthur Conmy, Buck Shlegeris, and Jacob Steinhardt. 2022. Interpretability in the wild: a circuit for indirect object identification in gpt-2 small. *Preprint*, arXiv:2211.00593.

A Related Literature

Mechanisms of Acquisition and Release. Grokking is characterized by a delayed transition from memorization to generalization (Power et al., 2022), while catastrophic forgetting describes the abrupt loss of prior knowledge during sequential training (McCloskey and Cohen, 1989). Traditional accounts focus on optimization dynamics or weight norms (Liu et al., 2023). However, Nanda et al. (2023) demonstrated that grokking is preceded by the gradual formation of "circuits"—computational subnetworks hidden from validation accuracy. We build upon the metric formalized by Sun (2025), using *Circuit Stability* to demonstrate that both acquisition and release are unified by the structural consistency of these internal subnetworks. This allows us to characterize grokking as the emergence of a stable circuit and forgetting as its eventual fragmentation.

Representational Drift vs. Structural Integrity. High-dimensional representational shift is often cited as the primary driver of forgetting (Kirkpatrick et al., 2017). Function Vectors (FVs) serve as latent triggers that compress task-specific knowledge into distilled activation patterns (Todd et al., 2024). While research suggests that regulating these vectors can influence model behavior (Jiang et al., 2025), our ablation results reveal a novel decoupling: task performance and reasoning can persist despite significant FV drift, provided the downstream circuit remains structurally invariant. This distinguishes our work from studies that treat representation and execution as synonymous, suggesting that the "functional origin" of a skill is tied more to its connectivity than its coordinate in activation space.

Selective Replay and Structural Anchoring. Mitigation of forgetting typically involves weight-space regularization like EWC (Kirkpatrick et al., 2017) or experience replay (Rolnick et al., 2019). However, these methods often impose global constraints that hinder the acquisition of new tasks. Our Stability-Guided Replay (SGR) addresses this by utilizing the stability metric from Sun (2025) as a high-fidelity filter to select "anchor" samples. By replaying data that exhibits the highest structural consistency, SGR anchors the model to its most robust reasoning pathways. This approach allows for the targeted preservation of specific circuits without the performance overhead or interference associated with global parameter constraints.

B Proof of Proposition 4.1

Proof. Denote $\delta_v(x) \triangleq {r_v}^{(t)}(x) - {r_v}^{(t')}(x)$ and $\delta(x) := f_t(x) - f_{t'}(x) = \sum_{v \in \mathcal{V}_{f_t}} \delta_v(x)$ by Equation 1. From Equation 2,

$$v^{(t)} - v^{(t')} = \Big(\mathbb{E}_{\mathcal{D}_1}[\delta(x)] - \mathbb{E}_{\mathcal{D}_0}[\delta(x)]\Big).$$

By triangle inequality,

$$\|v^{(t)} - v^{(t')}\|_2 \le \|\mathbb{E}_{\mathcal{D}_1}[\delta(x)]\|_2 + \|\mathbb{E}_{\mathcal{D}_0}[\delta(x)]\|_2.$$

Now we analyze either terms separately. Fix $j \in \{0, 1\}$. Using the fact that f_t admits an additive residual decomposition and triangle inequality we have,

$$\|\mathbb{E}_{\mathcal{D}_j}[\delta(x)]\|_2 = \left\| \sum_{v \in \mathcal{V}_{f_t}} \mathbb{E}_{\mathcal{D}_j}[\delta_v(x)] \right\|_2 \le \sum_{v \in \mathcal{V}_{f_t}} \|\mathbb{E}_{\mathcal{D}_j}[\delta_v(x)]\|_2.$$

By Jensen's Inequality since $\|\cdot\|_2$ is convex, $\|\mathbb{E}[\delta_v]\|_2 \le \mathbb{E}[\|\delta_v\|_2]$. Then, by the Cauchy-Schwarz Inequality, $\mathbb{E}[\|\delta_v\|_2] \le \sqrt{\mathbb{E}[\|\delta_v\|_2^2]}$. And so,

$$\|\mathbb{E}_{\mathcal{D}_j}[\delta(x)]\|_2 \le \sum_{v \in \mathcal{V}_{f_t}} \sqrt{\mathbb{E}_{\mathcal{D}_j}[\|\delta_v(x)\|_2^2]} \le \sqrt{|\mathcal{V}_{f_t}|} \left(\sum_{v \in \mathcal{V}_{f_t}} \mathbb{E}_{\mathcal{D}_j}[\|\delta_v(x)\|_2^2] \right)^{1/2}.$$

Applying this bound for $j = 1$ and $j = 0$ and using Equation 3 yields

$$\|v^{(t)} - v^{(t')}\|_2 \le \sqrt{|\mathcal{V}_{f_t}|} \Big(\sum_v \mathbb{E}_{\mathcal{D}_1} \|\delta_v\|_2^2 \Big)^{1/2} + \sqrt{|\mathcal{V}_{f_t}|} \Big(\sum_v \mathbb{E}_{\mathcal{D}_0} \|\delta_v\|_2^2 \Big)^{1/2} \le 2\sqrt{|\mathcal{V}_{f_t}|}\, \Delta(t, t'),$$

which implies equation 4. If $\Delta(t, t') = 0$, then $v^{(t)} = v^{(t')}$.

Now we show that the converse does not hold. Let $\mathcal{V}_{f_t} = \{1, 2\}$, take any nonzero $u \in \mathbb{R}^d$, and define (for all x)

$$r_1^{(t)}(x) = u, \quad r_2^{(t)}(x) = 0, \qquad r_1^{(t')}(x) = 0, \quad r_2^{(t')}(x) = u.$$

Then $f_t(x) = u = f_{t'}(x)$, so $v^{(t)} = v^{(t')}$ by Equation 2 for any $\mathcal{D}_0, \mathcal{D}_1$. However, $\|r_1^{(t)} - r_1^{(t')}\|_2^2 = \|u\|_2^2$ and $\|r_2^{(t)} - r_2^{(t')}\|_2^2 = \|u\|_2^2$, so Equation 3 gives $\Delta(t, t')^2 = 2\|u\|_2^2 > 0$. Thus FV stability can hold while circuit drift is arbitrarily large. □

C Implementation Details and Hyperparameters

C.1 Fine-Tuning Curriculum and Dataset Composition

Our sequential learning curriculum consists of four distinct tasks processed in a fixed order: BoolQ (T_1), MMLU-Computer Science (T_2), Winogrande (T_3), and StereoSet (T_4). For each task, we construct a question-answering format designed to elicit clear functional responses from the model.

- **BoolQ:** Boolean questions requiring a "Yes" or "No" response based on a provided passage.
- **MMLU-CS:** Multiple-choice questions (A/B/C/D) focusing on high-level computer science concepts.
- **Winogrande:** Commonsense reasoning tasks requiring the selection of a correct referent for a pronoun.
- **StereoSet:** Intrasentence bias evaluation tasks formatted as a choice between stereotypical and anti-stereotypical completions.

C.2 Training Configuration

The training was conducted using the `Llama-3.2-1B` architecture. We utilize the AdamW optimizer with a linear learning rate schedule. To isolate the effects of structural drift during task transitions, we save the model state after the completion of each task, denoted as milestones M_1 through M_4.

Specific hyperparameters utilized across all experiments are detailed in Table 4.

Table 4: Hyperparameter configurations for sequential fine-tuning.

Hyperparameter	**Value**
Base Model	Llama-3.2-1B
Optimizer	AdamW
Learning Rate	5×10^{-5}
Weight Decay	0.01
Batch Size	4
Epochs per Task	1
Max Sequence Length	512 tokens
Precision	BFloat16
Hardware	1 × NVIDIA A100 (80GB)

C.3 Answer Extraction and Evaluation

To calculate accuracy, we utilize a strict regex-based extraction pipeline as implemented in our `QATrainer`. The model is prompted to provide a concise answer at the end of the input sequence. For multiple-choice tasks (MMLU, Winogrande), we extract the first valid option identifier; for BoolQ, we normalize outputs to binary classes. Accuracy is defined as the ratio of correctly extracted answers to the total number of samples in the validation split for that milestone.

D STABILITY-GUIDED REPLAY (SGR) DETAILS

D.1 SGR PSEUDOCODE

The SGR algorithm differentiates itself from standard Experience Replay (ER) by using structural consistency as a selection filter. Rather than uniform sampling, it populates a buffer $\mathcal{B}$ only with samples that elicit stable, rank-invariant circuits. The training loop follows a "Task-then-Replay" sequence to prioritize structural anchoring.

Input: Model θ, Task Sequence $\{T_1, \ldots, T_n\}$, Stability Threshold $\tau = 0.5$, Buffer $\mathcal{B}$
for *each Task* T_i **do**
  `// Phase 1: Standard Fine-tuning`
  Train θ on T_i for one epoch using $\mathcal{L}_{clm}$
  Save temporary checkpoint θ_{temp}
  `// Phase 2: Stability Evaluation`
  for *each sample* $x \in T_i$ **do**
    Compute stability score S_x via edge-importance rank correlation
    if $S_x \geq \tau$ **then**
      Add x to $\mathcal{B}$
    end
  end
  `// Phase 3: Structural Anchoring (Replay)`
  if $\mathcal{B}$ *is not empty* **then**
    Sample batch $x_{replay} \sim \mathcal{B}$
    Fine-tune θ on x_{replay} to stabilize previous circuits
  end
end

Algorithm 1: Stability-Guided Replay (SGR)

D.2 SGR HYPERPARAMETERS

Table 5 outlines the specific configurations used for the stability-guided selection and replay phases. The stability threshold τ was determined empirically to isolate the top-tier invariant "load-bearing" samples.

Table 5: Hyperparameters for SGR selection and buffer management.

Parameter	**Value**
Stability Metric	Spearman Rank Correlation of Edges
Stability Threshold (τ)	0.5
Buffer Capacity	10,000 samples
Replay Batch Size	4
Replay Learning Rate	5×10^{-5}

E ELASTIC WEIGHT CONSOLIDATION (EWC) IMPLEMENTATION

E.1 EWC FORMULATION

As a parameter-level baseline, we implement Elastic Weight Consolidation (EWC) to compare structural circuit stability against weight-space regularization. For each task T_i, we compute the diagonal of the Fisher Information Matrix F, which represents the importance of each parameter θ_j to the task's performance. The loss function for a subsequent task T_{i+1} is defined as:

$$\mathcal{L}(\theta) = \mathcal{L}_{\text{clm}}(\theta) + \sum_{k<i} \frac{\lambda_{\text{ewc}}}{2} F_k(\theta - \theta_k^*)^2 \tag{5}$$

where θ_k^* are the optimized parameters for previous tasks and λ_{ewc} is the regularization strength.

E.2 EWC HYPERPARAMETERS

The EWC baseline was trained using the same base architecture (Llama-3.2-1B) and learning rate as SGR to ensure a controlled comparison. The specific regularization parameters used in our implementation are detailed in Table 6.

Table 6: Hyperparameters for the EWC baseline.

Parameter	**Value**
Regularization Weight (λ_{ewc})	400.0
Fisher Sample Size	Full Training Set
Fisher Computation	Diagonal Approximation
Optimizer	AdamW
Learning Rate	5×10^{-5}
Batch Size	2
Precision	BFloat16

E.3 FISHER INFORMATION CALCULATION

To compute the Fisher Information Matrix, we perform a dedicated pass over the task's dataset after the initial fine-tuning phase. For each sample, we calculate the squared gradients of the log-likelihood:

$$F = \mathbb{E}\left[\left(\nabla_\theta \log P(y|x,\theta)\right)^2\right] \tag{6}$$

These values are accumulated and normalized to provide the per-parameter importance weighting used during the sequential training of subsequent milestones.

F EVALUATION METRICS AND DATASET STATISTICS

F.1 EVALUATION METHODOLOGY

Performance is evaluated using two primary metrics: Accuracy (A) and Circuit Stability (S).

- **Accuracy (A):** Calculated using a strict regex-based extraction of the model's generated completion. For multiple-choice tasks, we verify the presence of the correct option identifier (e.g., "(A)"); for boolean tasks, we normalize responses to binary classes.
- **Circuit Stability (S):** Measured as the Spearman rank-correlation of edge importance scores derived from two disjoint partitions of the validation set ($D_{\text{val},1}$ and $D_{\text{val},2}$). Edge importance is computed via the gradient-based attribution method defined in Section 2.

F.2 DATASET STATISTICS

We evaluate our hypothesis across two distinct task orders to ensure the relationship between S and forgetting is not sequence-dependent: Sequence 1 ($T_1 \to T_2 \to T_3 \to T_4$) and Sequence 2 ($T_4 \to T_3 \to T_2 \to T_1$). The composition of these datasets is detailed in Table 7.

Table 7: Statistics for the datasets used in the sequential fine-tuning curriculum.

Task ID	**Dataset**	**Training Samples**	**Validation Samples**
T_1	BoolQ	9,427	3,270
T_2	MMLU-CS	100*	100
T_3	Winogrande	40,398	1,767
T_4	StereoSet†	2,106	2,106

*MMLU-CS utilizes the dev/test split as standard.

†StereoSet lacks a train split; the validation set is used for both training and evaluation.

G GROKKING ANALYSIS: EXPERIMENTAL DETAILS

G.1 GSM-SYMBOLIC DATASET CONFIGURATION

To evaluate the relationship between structural stability and generalization, we utilize the GSM-Symbolic benchmark. This dataset provides a controlled environment to verify if a model has "grokked" underlying mathematical logic rather than memorizing templates.

- **Procedural Generation:** In GSM-Symbolic (Mirzadeh et al., 2025), symbolic templates are used to generate mathematical word problems. This ensures that while the reasoning chain remains procedurally consistent, the numerical values and linguistic framing in the validation set are entirely disjoint from the training data.
- **Standardization:** All problems are formatted as Question-Answer pairs. The model is trained to generate the full reasoning path, with accuracy determined by the final numerical value extracted via regex.

G.2 TRAINING AND MONITORING PROTOCOL

The model is fine-tuned using the parameters detailed in Table 8. We monitor the structural state of the model via a high-frequency evaluation loop every 2,000 steps.

Table 8: Hyperparameters for Grokking Analysis on GSM-Symbolic.

Parameter	**Value**
Learning Rate	5×10^{-5}
Batch Size	2
Evaluation Frequency	Every 2,000 steps
Weight Decay	0.01
Optimizer	AdamW
Stability Evaluation Split	Validation (Symbolic-Disjoint)

G.3 GROKKING EVENT DETECTION AND STABILITY CORRELATION

We define a "Grokking Event" as a rapid surge in validation accuracy ($\geq 5\%$ within 2,000 steps) occurring after training accuracy has already saturated. For each milestone, we compute:

1. **Validation Accuracy:** Performance on unseen symbolic variations.
2. **Circuit Stability (S):** The Spearman rank-correlation of edge importance scores across disjoint partitions of the validation set.

H ABLATION METHODOLOGY: FV AND STABILITY DECOUPLING

To decouple the effects of latent triggers (Function Vectors) and structural pathways (Circuits), we implemented two specific intervention strategies within the `ContinualAblationQATrainer`.

H.1 SETTING 1: FIXED FV VIA INTERVENTION

In the "Fixed FV / Disrupted Circuit" experiment, we manually prevent the Function Vector from drifting while allowing the model parameters to undergo standard fine-tuning.

- **Implementation:** We use a forward hook on the causal attention heads identified as part of the T_1 function vector. During every forward pass of subsequent tasks (T_2, T_3, T_4), this hook intercepts the mean-head activations.
- **Activation Patching:** The activations are overwritten with the "gold" activations $\mathbf{v}_{M1}$ recorded at the end of Task 1. This ensures that the model always perceives the original latent trigger, even as the weights W are modified by standard $\mathcal{L}_{\text{clm}}$ gradient updates.

- **Methodological Justification:** We chose direct activation patching over alternative methods, such as normalizing drift by its magnitude within the loss function (e.g., a penalty term $\frac{\|\mathbf{v}_t - \mathbf{v}_{\text{init}}\|}{\|\mathbf{v}_{\text{init}}\|}$). While normalization accounts for the relative scale of representation shifts, it remains a "soft" constraint that allows the model to find degenerate solutions where the vector orientation shifts while the magnitude remains constant. By using a "hard" hook to fix the vector in both direction and magnitude, we ensure that any observed forgetting is definitively caused by the fragmentation of the downstream circuit rather than subtle representational drift that a normalized penalty might miss.

H.2 Setting 2: Enforced Stability via Penalty

In the "Allowed FV Drift / Enforced Circuit" experiment, we do not intervene in the forward pass, allowing the activations to drift. Instead, we modify the optimization objective to preserve the circuit topology.

- **Stability Penalty:** We introduce a structural regularization term $\mathcal{L}_{\text{stab}} = \lambda(S_{\text{curr}} - S_{\text{init}})^2$, where S is the circuit stability score. In the implementation, λ is set to a high constant to prioritize structural preservation.
- **Gradient Dynamics:** The total loss $\mathcal{L} = \mathcal{L}_{\text{clm}} + \mathcal{L}_{\text{stab}}$ forces the optimizer to find parameter updates that reside in the intersection of the new task's manifold and the previous task's circuit ranking.

H.3 FV Drift Measurement

Drift (d_{init}) is calculated post-epoch. We run a reference pass over T_1 data using the current milestone checkpoint to extract the new mean-head activations $\mathbf{v}_t$. The drift is the L_2 norm $\|\mathbf{v}_t - \mathbf{v}_{M1}\|_2$. This measurement is independent of the training hooks and serves as the primary metric for quantifying representational shift.

I Circuit Stability-Guided Replay (SGR)

I.1 Methodology and Baselines

To mitigate the structural fragmentation identified in Section 4, we introduce **Circuit Stability-Guided Replay (SGR)**. SGR curates a replay buffer based on the invariance of a model's internal reasoning pathways rather than raw input features.

SGR Algorithm. Following the partitioning framework of Sun (2025), SGR populates a replay buffer with samples exceeding a stability threshold $\tau = 0.5$, representing the "load-bearing" structural components of a task's circuit. Training follows a "task-then-replay" sequence: the model completes an epoch on the current task T_i before performing a replay phase on previous high-stability samples. This replay phase acts as a structural anchor; by prioritizing the preservation of invariant reasoning pathways over representational drift, it trades peak current-task optimization for long-term circuit stability. Pseudocode and settings are provided in Appendix D.

Baselines. We compare SGR against three standard approaches:

- Vanilla: Sequential fine-tuning with no retention strategy.
- Elastic Weight Consolidation (EWC): A regularization method that penalizes shifts in parameters critical to previous tasks by calculating a Fisher Information Matrix (Kirkpatrick et al., 2017). See Appendix E for the implementation of the EWC trainer.
- Uniform Replay: A buffer-based method that samples past data with equal probability, regardless of the model's internal state or the sample's difficulty.

I.2 Comparative Analysis of Retention

SGR Performance. As seen in Table 9, SGR is the only method that prevents total functional collapse for the earliest task in the sequence. While all baselines drop to 0.00 accuracy for BoolQ

Table 9: **Comparative Accuracy Suite for Sequence 1.** Values denote validation accuracy at checkpoints M_1–M_4. SGR demonstrates superior retention for early tasks (T_1, T_2) while maintaining performance on subsequent tasks compared to parameter-level regularization (EWC) and indiscriminate data replay (Uniform).

Milestone	Method	T_1 (BoolQ)	T_2 (MMLU)	T_3 (Wino)	T_4 (Stereo)
M_1	Vanilla	0.6232	0.0000	0.0000	0.0000
	EWC	**0.7644**	0.0000	0.0000	0.0000
	Uniform	0.6221	0.0000	0.0000	0.0000
	SGR (Ours)	0.6537	0.0000	0.0000	0.0000
M_2	Vanilla	0.0000	**0.5556**	0.0000	0.0000
	EWC	0.0000	**0.5556**	0.0000	0.0000
	Uniform	0.6221	0.1667	0.0000	0.0000
	SGR (Ours)	**0.6537**	0.1111	0.0000	0.0000
M_3	Vanilla	0.0000	0.1111	0.5043	0.0000
	EWC	0.0000	**0.1667**	0.5020	0.0000
	Uniform	**0.6221**	0.0000	**0.5422**	0.0000
	SGR (Ours)	0.6101	0.1111	0.5043	0.0000
M_4	Vanilla	0.0000	0.0000	0.0489	0.3310
	EWC	0.0000	0.0000	0.0805	0.3281
	Uniform	0.0000	0.0000	0.4963	**0.3333**
	SGR (Ours)	**0.3785**	**0.1111**	**0.5043**	**0.3333**

(T_1) by the final milestone (M_4), SGR retains an accuracy of 0.3785. Furthermore, SGR maintains the highest consistency for MMLU (T_2), preserving 0.1111 accuracy throughout M_2 to M_4, whereas Uniform Replay fails to retain any T_2 capability by M_3.

Structural Rigidity in EWC. EWC demonstrates the poorest performance among the retention strategies. While it achieves the highest initial accuracy for T_1 (0.7644) at M_1, it suffers an immediate collapse to 0.00 at M_2. Furthermore, EWC fails to effectively learn T_3 (0.0805 at M_4), suggesting that parameter-level constraints induce a structural rigidity that inhibits the acquisition of new tasks.

Efficiency of Stability Selection. Uniform Replay maintains T_1 accuracy through M_3 (0.6221) but fails catastrophically at M_4. This indicates that indiscriminate sampling eventually fails to protect the underlying circuit as the model's total capacity is exhausted. By contrast, SGR's selective approach—prioritizing samples that anchor stable circuits—provides a more robust defense against fragmentation. At M_4, SGR matches the current task performance of all baselines ($T_4 \approx 0.33$) but does so while preserving 60.8% of its peak T_1 accuracy. This supports our hypothesis that reinforcing structurally invariant sub-networks is a more memory-efficient strategy for continual learning than parameter regularization or uniform data replay.